Magda's notes about **compression**

Last update: June 4, 2021

Informal notes for my future self who is likely to forget. I explain the papers the way I understand them, using terminology and logic natural to me. This means I may deviate from the original paper structure, notation, etc. At places, my interpretation my be incorrect due to lack of understanding. I will strive for this not to happen too often but I'm certainly not infallible.

This is a working document, not polished, with possible typos, editing errors, etc.

Contents

1 Intro thougths

maximum likelihood we assume a population $\mathbf{x} \in \mathbb{X}$ governed by some unknown probablity distribution $p(\mathbf{x})$. We have at our disposal a sample from this population $\mathbb{S} = \{\boldsymbol{x}_i\}_{i=1}^n$ (assume i.i.d.) and we use it to infer a statistical model of the data. In maximum liekelihood approach we posit a distribution family $q(\mathbf{x}|\boldsymbol{\theta})$ and we search for the best parameters $\boldsymbol{\theta}^*$ so that the distribution $q(\mathbf{x}|\boldsymbol{\theta}^*)$ best fits the observed data $\sim$ we search parameters that would have most likely produced the data.

The max *likelihood* problem is thus $\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \prod_i^n q(\boldsymbol{x}_i|\boldsymbol{\theta})$ which is equivalent to minimizing the negative log of the likelihood

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} - \sum_i^n \log q(\boldsymbol{x}_i|\boldsymbol{\theta}), \quad \boldsymbol{x}_i \sim p(\mathbf{x}) \ . \tag{1.1}$$

compression the theoretical lower bound (thanks to Shanon) on the expected length of a binary code for random variable $\mathbf{x}$ is the entropy of the distribution $H(p) = -E_{p(\mathbf{x})} \log_2 p(\mathbf{x})$, where each sample $\boldsymbol{x}$ is encoded with the shortest possible binary code of length equal to its information content $h(\boldsymbol{x}) = -\log_2 p(\boldsymbol{x})$.

Since the true data distribution $p(\mathbf{x})$ is unknown, we cannot use it for compression. We use instead a compression model, a distribution $q(\mathbf{x}|\boldsymbol{\theta})$, to establish the lengths of the codewords for the symbols $\boldsymbol{x}$ as $l(\mathbf{x}) = -\log_2 q(\boldsymbol{x}|\boldsymbol{\theta})$. These lengths are clearly not the same as the information content of the individual observations $\boldsymbol{x}$. The expected codeword length for a randomly sampled symbol $\mathbf{x}$ is the cross-entropy $CH(p,q) = -E_{p(\mathbf{x})} \log_2 q(\mathbf{x}|\boldsymbol{\theta})$.

To minimize the expected codeword length of symbols sampled from the unknown $p(\mathbf{x})$, we shall minimize the cross-entropy $\boldsymbol{\theta}^* = \arg\min_\theta -E_{p(\mathbf{x})} \log_2 q(\mathbf{x}|\boldsymbol{\theta})$. In practice, this can be achieved by minimizing the empirical estimate over the data sample

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} - \frac{1}{n} \sum_i^n \log_2 q(\boldsymbol{x}_i|\boldsymbol{\theta}), \quad \boldsymbol{x}_i \sim p(\mathbf{x}) \ . \tag{1.2}$$

Note that this is equialent to problem (1.1).

> Maximizing likelihood is equivalent to finding a compression model that will minimize the expected codeword length.

Furthermore we can develop the cross-entropy as follows

$$\begin{aligned} CH(p,q) &= -E_{p(\mathbf{x})} \log_2 q(\mathbf{x}|\boldsymbol{\theta}) \\ &= E_{p(\mathbf{x})} \log_2 \frac{p(\mathbf{x})}{q(\mathbf{x}|\boldsymbol{\theta})p(\mathbf{x})} \\ &= E_{p(\mathbf{x})} \log_2 \frac{p(\mathbf{x})}{q(\mathbf{x}|\boldsymbol{\theta})} - E_{p(\mathbf{x})} \log_2 p(\mathbf{x}) \\ &= D_{\mathrm{KL}}(p,q) + H(p) \ . \end{aligned} \tag{1.3}$$

Since the unkown p and hence the entropy $H(p)$ are fixed, minimizing the cross entropy $H(p,q)$ is also equivalent to minimizing the KL divergence between p and q which can be interpreted as

- minimizing the statistical distance between the distribution (bringing q close to p)
- minimizing the additional bits from using q instead of p to establish the codewords.

2 Freay and Hinton: Bits-back with arithmetic coding

Paper: Brendan J. Frey and Geoffrey E. Hinton. "Efficient Stochastic Source Coding and an Application to a Bayesian Network Source Model". In: *Computer Journal* 40.2/3 (1997), pp. 157–165

Notes taken: 28/2/2021

TLDR: Learn the transformation

2.1 Intro

A *source code* uses as *source model* to map each input symbol to a unique *codeword*. There are two principal source models: **one-to-one** where each symbol is mapped to a single codeword; **one-to-many** where each symbol is mapped to a distribution across multiple codewords. One-to-many source models arrise naturally in many domains such as in mixture distributions

$$\mathbf{x} \in \mathbb{X} \sim p(\mathbf{x}) = \sum_y p(\mathbf{y}) p(\mathbf{x}|\mathbf{y}) \quad . \tag{2.1}$$

.

For example for the mixture of Gaussian in figure 1 the optimal one-to-one model uses codewords with length given by the information content of the symbols $l(\mathbf{x}) = h(\mathbf{x}) = -\log_2 p(\mathbf{x})$.

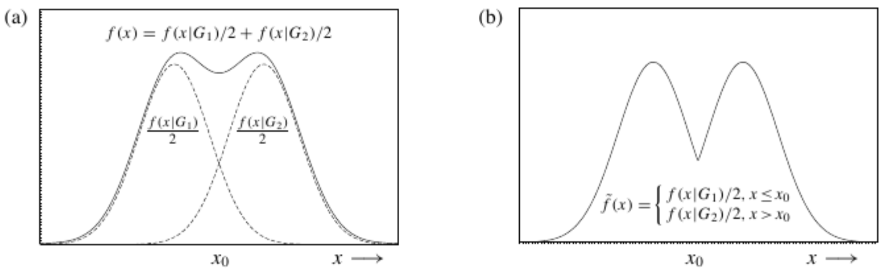

Figure 1: Mixture of two Guassians.

In one-to-many model, we have for each symbol $\boldsymbol{x}$ two possible codewords: one given by G_1 the other by G_2 with the lengths given by $h_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}) = -\log_2 p(\mathbf{x}|\mathbf{y} = \mathrm{y}), \mathrm{y} \in \{1, 2\}$ respecitvely. Assuming the prior $p(\mathbf{y} = 1) = p(\mathbf{y} = 2) = 0.5$ we also have $h_{\boldsymbol{y}}(1) = h_{\boldsymbol{y}}(2) = -\log_2 0.5 = 1$ bit to communicate $\mathbf{y}$ and hence the identity of the conditional distribution.

A natural choice is to always pick the shorter of the two codewords for $\boldsymbol{x}$. This corresponds to a distribution in (b) in figure 1 which is clearly suboptimal (gives longer codes then the one-to-one) around $\boldsymbol{x}_0$.

Here they show how stochastic one-to-many model relying on the *bits-back* idea can achieve same rates as the one-to-one scheme.

Running example: We want to encode a symbol $\boldsymbol{x}$ that is twice as likely under G_1 then G_2 so that it requires 2 and 3 bits respectively under the two distributions (that is $p(\boldsymbol{x}|\mathbf{y} = 1) = 1/4$ and $p(\boldsymbol{x}|\mathbf{y} = 2) = 1/8$). With 1 bit to specify the $\mathbf{y}$ and hence the conditional distribution we get the total lenght of 3 and 4 bits for the two-level code. We now have the following options:

1. picking alwyas the shorter code gives expected length of 3 bits;
2. picking between the two codes equally often (according to the prior $p(\mathbf{y})$) gives expected length of 3.5 bits;
3. use bits-back to get the expected length 2.5 bits.

This is the bits-back idea: Instead of picking always the shortest codeoword or pick equally often amongst the codeword in the one-to-many setup, you will be picking $\boldsymbol{y} \in 1, 2$ randomly according to some probability $q(\mathbf{y}|\mathbf{x})$. However, to select (sample) $\mathbf{y}$ you will use some random sampler based on transorming some auxiliary data $\mathbf{z}$ into the samles $\mathbf{y} \sim q(\mathbf{y})$. A trivial example for univariate case is the inverse CDF sampling starting from uniform $\mathbf{z}$ or, as they suggest here, the arithmetic decoding which can produce vectorial $\mathbf{y}$ starting from a random binary sequence $\mathbf{z}$. Once you sampled $\boldsymbol{y}$ you will encode it using $p(\mathbf{y})$ and concatenate with the encoding of $\boldsymbol{x}$ using the corresponding $p(\boldsymbol{x}|\boldsymbol{y})$. The message you transmit thus has length $l(\boldsymbol{x}, \boldsymbol{y}) = -\log_2 p(\boldsymbol{x}|\boldsymbol{y}) - \log_2 p(\boldsymbol{y})$.
The reciever has access to the encoding models $p(\mathbf{y}), p(\mathbf{x}|\mathbf{y})$ and it can thus easily decode the symbols $\mathbf{x}$ and $\mathbf{y}$ from the codewords $C(\mathbf{x}, \mathbf{y})$. However, it also has access to the sampling distribution $q(\mathbf{y})$ and the algirithm the encoder uses to sample $\mathbf{y}$ by transforming $\mathbf{z}$. It can thus reverse this operation (so the transformation must be one-to-one) and recover $\mathbf{z}$ at no additional communication cost - these are the 'bits-back'. If $\mathbf{z}$ contains some useful information, these 'bits-back' essentially reduce the *effective message length*.

In our **running example** lets imagine $\mathbf{z} \in a, b$ with $p(\mathbf{z} = a) = p(\mathbf{z} = b) = 0.5$. Then once we recover $\mathbf{z}$ we have communicated $h_{\mathbf{z}}(\boldsymbol{z}) = -\log_2 0.5 = 1$ bit of information at no cost and so the **expected effective length is** $3.5 - 1 = 2.5$ **bits**.

How many bits can be recovered is a function of the probability distribution $q(\mathbf{y}|\mathbf{x})$.

The **average code length** for a symol $\boldsymbol{x}$ is

$$\mathcal{E}(\boldsymbol{x}) = \sum_{y} q(\boldsymbol{y}|\boldsymbol{x}) l(\boldsymbol{x}, \boldsymbol{y}) \ . \tag{2.2}$$

Here $l(\boldsymbol{x}, \boldsymbol{y}) = -\log_2 p(\boldsymbol{x}|\boldsymbol{y}) - \log_2 p(\boldsymbol{y})$ are the different lengths using different conditionals to encode $\boldsymbol{x}$ depending on the choice of $\boldsymbol{y}$.

The average information content (the entropy) of the selecting distribution

$$\mathcal{H}(\boldsymbol{x}) = -\sum_{y} q(\boldsymbol{y}|\boldsymbol{x}) \log_2 q(\boldsymbol{y}|\boldsymbol{x}) \tag{2.3}$$

are **the average bits back we can recover** for this symbol $\mathbf{x}$ when choosing $\boldsymbol{y}$ according to $q(\mathbf{y}|\boldsymbol{x})$.

This may seem strange cause the symbols we will recover are $\mathbf{z}$, not $\mathbf{y}$ again. But since the sampling transformation $f : \mathbf{z} \to \mathbf{y}$ has to be one-to-one and for the moment **we assume all the variables are discrete** we heve the trivial result due to prservation of the probability that $q(f(\mathbf{z})) = q(\mathbf{y}|\boldsymbol{x})$ (the only thing that happens is re-labelling of the variables) and that the entropy of these two distributions is equal

$$\mathcal{H}(\mathbf{x}) = -\sum_{y} q(\boldsymbol{y}|\boldsymbol{x}) \log_2 q(\boldsymbol{y}|\boldsymbol{x}) = -\sum_{z} q(\boldsymbol{z}) \log_2 q(\boldsymbol{z}) \tag{2.4}$$

A strange consequence of this that **I don't yet understand** is that the sampling distribution of $q(\mathbf{z})$ essentially has to be the same as $q(\mathbf{y}|\mathbf{x})$. So the inverse CDF above is probably not a good example. But this gives potentially room for research :)

The average effective codeword length of symbol $\boldsymbol{x}$ is the difference between the two above

$$\mathcal{F}(\boldsymbol{x}) = \mathcal{E}(\boldsymbol{x}) - \mathcal{H}(\boldsymbol{x}) \ . \tag{2.5}$$

This is also called the ***free energy*** - same concept known from statistical physics.

Next comes the question what $q(\mathbf{y}|\boldsymbol{x})$ shall be. They claim it shall be the Boltzman distribution based on the codeword lengths

$$q^*(\boldsymbol{y}|\boldsymbol{x}) = \frac{2^{-l(\boldsymbol{x},\boldsymbol{y})}}{\sum_{\boldsymbol{y}'} 2^{-l(\boldsymbol{x},\boldsymbol{y}')}} \tag{2.6}$$

My proof for Boltzman:

$$\begin{aligned}
\mathcal{F}^*(\boldsymbol{x}) = \mathcal{E}^*(\boldsymbol{x}) - \mathcal{H}^*(\boldsymbol{x}) &= \sum_y q^*(\boldsymbol{y}|\boldsymbol{x}) l(\boldsymbol{x},\boldsymbol{y}) + \sum_y q^*(\boldsymbol{y}|\boldsymbol{x}) \log_2 q^*(\boldsymbol{y}|\boldsymbol{x}) \\
&= \sum_y \frac{2^{-l(\mathbf{x},\mathbf{y})}}{\sum_{y'} 2^{-l(\mathbf{x},\mathbf{y}')}} \left[l(\mathbf{x},\mathbf{y}) + \log_2 \frac{2^{-l(\mathbf{x},\mathbf{y})}}{\sum_{y'} 2^{-l(\mathbf{x},\mathbf{y}')}} \right] = l(\mathbf{x},\mathbf{y}) - l(\mathbf{x},\mathbf{y}) - \log_2 \sum_{y'} 2^{-l(\mathbf{x},\mathbf{y}')} \\
&= -\log_2 \sum_{y'} 2^{-l(\mathbf{x},\mathbf{y}')} = -\log_2 \sum_{y'} 2^{\log_2 p(\boldsymbol{x},\boldsymbol{y})} = -\log_2 \sum_{y'} p(\boldsymbol{x},\boldsymbol{y}) = -\log_2 p(\boldsymbol{x})
\end{aligned}$$

The Boltzman brings the free energy to the same lengths as if one-to-one code with $p(\mathbf{x})$ were used so it is optimal. $\square$

I'd think it shall be the posterior $p(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y})p(\boldsymbol{x}|\boldsymbol{y})/p(\boldsymbol{x})$.

My proof for posterior:

$$\begin{aligned}
\mathcal{F}^*(\boldsymbol{x}) = \mathcal{E}^*(\boldsymbol{x}) - \mathcal{H}^*(\boldsymbol{x}) &= \sum_y p(\boldsymbol{y}|\boldsymbol{x}) l(\boldsymbol{x},\boldsymbol{y}) + \sum_y p(\boldsymbol{y}|\boldsymbol{x}) \log_2 p(\boldsymbol{y}|\boldsymbol{x}) \\
&= \sum_y p(\boldsymbol{y}|\boldsymbol{x}) \left[-\log_2 p(\boldsymbol{x},\boldsymbol{y}) + \log_2 p(\boldsymbol{y}|\boldsymbol{x}) \right] = \sum_y p(\boldsymbol{y}|\boldsymbol{x}) \left[-\log_2 p(\boldsymbol{x}) \right] = -\log_2 p(\boldsymbol{x})
\end{aligned}$$

The posterior brings the free energy to the same lengths as if one-to-one code with $p(\mathbf{x})$ were used so it is also optimal.

Does this mean that the Boltzman and the posterior distributions are the same?

$$\begin{aligned}
q^*(\boldsymbol{y}|\boldsymbol{x}) &= \frac{2^{-l(\boldsymbol{x},\boldsymbol{y})}}{\sum_{\boldsymbol{y}'} 2^{-l(\boldsymbol{x},\boldsymbol{y}')}} = \frac{2^{\log_2 p(\boldsymbol{x},\boldsymbol{y})}}{\sum_{\boldsymbol{y}'} 2^{\log_2 p(\boldsymbol{x},\boldsymbol{y}')}} \\
&= \frac{p(\boldsymbol{x},\boldsymbol{y})}{\sum_{\boldsymbol{y}'} p(\boldsymbol{x},\boldsymbol{y}')} = \frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})} = p(\boldsymbol{y}|\boldsymbol{y})
\end{aligned}$$

So Boltzman baseded on the codeword lengths and the posterior are indeed one and the same thing!

$p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) = p(\mathbf{x},\mathbf{y})$ so if we use this to get the codewords we get the lengths $l(\mathbf{x},\mathbf{y}) = -\log_2 p(\mathbf{x},\mathbf{y})$. Plug this into the free energy

$$\begin{aligned}
\mathcal{F}(\mathbf{x}) = \mathcal{E}(\mathbf{x}) - \mathcal{H}(\mathbf{x}) &= \sum_y p(\mathbf{y}|\mathbf{x}) l(\mathbf{x},\mathbf{y}) + \sum_y p(\mathbf{y}|\mathbf{x}) \log_2 p(\mathbf{y}|\mathbf{x}) \\
&= \sum_y p(\mathbf{y}|\mathbf{x}) \left[-\log_2 p(\mathbf{x},\mathbf{y}) + \log_2 p(\mathbf{y}|\mathbf{x}) \right] = \sum_y p(\mathbf{y}|\mathbf{x}) \left[-\log_2 p(\mathbf{x}) \right] = -\log_2 p(\mathbf{x})
\end{aligned}$$

and we get this is equal to the optimal codelength for the one-to-one code based on $p(\mathbf{x})$. $\square$

In our **running example** we use the result for the optimal free energy $\mathcal{F}^*(\boldsymbol{x}) = -\log_2 \sum_{y'} 2^{-l(\mathbf{x},\mathbf{y}')}$ with the length $l(\boldsymbol{x}, 1) = 3$ bits and $l(\boldsymbol{x}, 2) = 4$ bits we get $\mathcal{F}^*(\boldsymbol{x}) = -\log_2(2^{-3} + 2^{-4}) = 2.415$ bits. This is shorter (better) then the naive sampling used above with $q(\mathbf{y}|\boldsymbol{x}) = 0.5$ which gave 2.5 bits.

Alternative ways to write the free energy

$$
\begin{aligned}
\mathcal{F}(\boldsymbol{x}) = \mathcal{E}(\boldsymbol{x}) - \mathcal{H}(\boldsymbol{x}) &= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) l(\boldsymbol{x}, \boldsymbol{y}) + \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 q(\boldsymbol{y}|\boldsymbol{x}) \\
&= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 2^{l(\boldsymbol{x},\boldsymbol{y})} + \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 q(\boldsymbol{y}|\boldsymbol{x}) \\
&= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{2^{-l(\boldsymbol{x},\boldsymbol{y})}} = \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{x}, \boldsymbol{y})} \\
&= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x}) p(\boldsymbol{x})} = \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x})} - \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 p(\boldsymbol{x}) \ .
\end{aligned}
$$

Here we are quickly getting to ELBO! Take the expected free energy

$$
\begin{aligned}
\mathbb{E}_{p(\mathbf{x})} \mathcal{F}(\mathbf{x}) &= \sum_x \sum_y p(\mathbf{x}) q(\boldsymbol{y}|\mathbf{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\mathbf{x})} - \sum_x \sum_y p(\boldsymbol{x}) q(\boldsymbol{y}|\boldsymbol{x}) \log_2 p(\boldsymbol{x}) \\
&= D_{\mathrm{KL}}(q(\mathbf{y}|\mathbf{x}) || p(\mathbf{y}|\mathbf{x})) - \mathbb{E}_{p(\mathbf{x})} \log_2 p(\mathbf{x}) = -ELBO
\end{aligned}
\tag{2.7}
$$

How much do we suffer from not using the Boltzman (posterior) $q^*(\boldsymbol{y}|\boldsymbol{x}) = p(\mathbf{y}|\boldsymbol{x})$ but some other distribution $q(\boldsymbol{y}|\boldsymbol{x})$ to sample $\mathbf{y}$? Its the difference in the free energies:

$$
\begin{aligned}
\mathcal{F}(\boldsymbol{x}) - \mathcal{F}^*(\boldsymbol{x}) &= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) l(\boldsymbol{x}, \boldsymbol{y}) + \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 q(\boldsymbol{y}|\boldsymbol{x}) + \log_2 p(\boldsymbol{x}) \\
&= \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x}) p(\boldsymbol{x})}{2^{-l(\boldsymbol{x},\boldsymbol{y})}} = \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x}) p(\boldsymbol{x})}{p(\boldsymbol{x}, \boldsymbol{y})} = \sum_y q(\boldsymbol{y}|\boldsymbol{x}) \log_2 \frac{q(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x})}
\end{aligned}
$$

This is the KL divergence between the sampling distribution $q(\boldsymbol{y}|\boldsymbol{x})$ and the optimal Boltzman or posterior distribution $p(\boldsymbol{y}|\boldsymbol{x}) = q^*(\boldsymbol{y}|\boldsymbol{x})$.

2.2 Bits-back coding algorithm

If $\mathbf{y}$ is not simply two-valued or diadic (powers of two) variable, they propose to use arithmetic coding as the $f : \mathbf{z} \to \mathbf{y}$ algorithm.

References

[1] Brendan J. Frey and Geoffrey E. Hinton. “Efficient Stochastic Source Coding and an Application to a Bayesian Network Source Model”. In: *Computer Journal* 40.2/3 (1997), pp. 157–165 (cit. on p. 3).

Index